CLAIMS

1.-16. (canceled)

17. (amended) A processor, comprising:

a plurality of registers;

instruction processing circuitry that fetches an instruction sequence for execution, said instruction sequence including a load instruction and a preceding instruction that precedes said load instruction in program order, wherein said instruction processing circuitry, after fetching said instruction sequence for execution and prior to dispatching said load instruction for execution and responsive to detecting said load instruction within said fetched instruction sequence, translates said load instruction into separately executable prefetch and register ~~operations~~ instructions and thereafter dispatches said prefetch and register operations for execution; ~~and~~

a request bus coupling to lower level memory; and

execution circuitry coupled to said request bus and coupled to receive dispatched instructions including said prefetch, register and preceding instructions from said instruction processing circuitry, wherein said execution circuitry ~~that performs~~ executes at least said prefetch instruction ~~operation~~ out-of-order with respect to said preceding instruction to prefetch data and subsequently separately executes said register instruction ~~operation~~ to place said data into a register among said plurality of registers specified by said load instruction, wherein said execution circuitry executes ~~performs~~ said prefetch instruction ~~operation~~ by calculating a speculative target memory address utilizing contents of at least one register identified by said prefetch instruction, without regard for whether said contents will be modified between calculation of said speculative target memory address and execution of ~~performing~~ said register instruction ~~operation~~, and by thereafter initiating a fetch, via said request bus, of said data from a memory location associated with[[in]] said speculative target memory address.

18. (amended) A method, said method comprising:

fetching an instruction sequence for execution, said instruction sequence including a load instruction and a preceding instruction that precedes said load instruction in program order;

in response to fetching said instruction sequence for execution and prior to execution of said load instruction, instruction processing circuitry detecting said load instruction within said fetched instruction sequence and translating said load instruction into separately executable prefetch and register instructions ~~operations~~;

after the translating, the instruction processing circuitry dispatching the preceding instruction and the prefetch and register instructions for execution;

execution circuitry receiving the dispatched preceding instruction and the prefetch and register instructions from the instruction processing circuitry;

in response to receiving the dispatched prefetch instruction, the execution circuitry executing ~~performing~~ at least said prefetch instruction ~~operation~~ out-of-order with respect to said preceding instruction to prefetch data, wherein executing ~~performing~~ said prefetch operation comprises:

the execution circuitry calculating a speculative target memory address utilizing contents of at least one register identified by the prefetch instruction without regard for whether said contents will be modified between calculation of said speculative target memory address and execution of ~~performing~~ said register instruction ~~operation~~; and

thereafter the execution circuitry initiating a fetch, via a request bus coupled to lower level memory, of said data from a memory location associated ~~within~~ said speculative target memory address; and

thereafter, the execution circuitry separately executing said register instruction ~~operation~~ to place said data into a register ~~among said plurality of registers~~ specified by said load instruction.

19. (amended) The processor of Claim 17, wherein said execution circuitry executes said register instruction ~~operation~~ in-order with respect to said preceding instruction.

20. (amended) The processor of Claim 17, wherein said execution circuitry executes said register instruction ~~operation~~ out-of-order with respect to said preceding instruction.

21. (amended) The processor of Claim 17, wherein said prefetch ~~operation~~ and ~~said~~ register instructions ~~operation~~ have a same operation code.

22. (amended) The processor of Claim 21, wherein said prefetch ~~operation~~ and ~~said~~ register ~~operation~~ instructions specify a same target register for said data and differ only in a value of a register operation field.

23. (amended) The processor of Claim 17, wherein said execution circuitry stores said data prefetched in response to said prefetch ~~operation~~ instruction in a temporary register.

24. (amended) The processor of Claim 17, and further comprising a data hazard detector that, in response to detection of a hazard for said data, signals said processor to discard said data and said register instruction ~~operation~~.

25. (amended) The method of Claim 18, and further comprising executing said register ~~operation~~ instruction in-order with respect to said preceding instruction.

26. (amended) The method of Claim 18, and further comprising executing said register ~~operation~~ instruction out-of-order with respect to said preceding instruction.

27. (amended) The method of Claim 18, wherein translating said load instruction comprises translating said load instruction into prefetch and register ~~operations~~ instructions having a same operation code.

28. (amended) The method of Claim 27, wherein said prefetch ~~operation~~ and ~~said~~ register ~~operation~~ instructions specify a same target register for said data and differ only in a value of a register operation field.

29. (amended) The method of Claim 18, wherein performing said prefetch instruction ~~operation~~ comprises storing said data in a temporary register.

30. (amended) The method of Claim 18, and further comprising:

detecting a data hazard for said data; and

in response to detection of said hazard for said data, discarding said data and said register instruction ~~operation~~.